

Low-carbon no-idle permutation flow shop scheduling problem: giant trevally optimizer vs African vultures optimization algorithm

Dana Marsetiya Utama, Cantika Febrita

Department of Industrial Engineering, Faculty of Engineering, Muhammadiyah Malang University, Malang, Indonesia

Article Info

Article history:

Received Apr 27, 2023

Revised Jun 26, 2023

Accepted Jul 15, 2023

Keywords:

African vultures optimization algorithm

Flow shop

Giant trevally optimizer

Low carbon

No-idle

ABSTRACT

Greenhouse gas emissions continue to increase due to increased energy consumption. One of the largest emission-contributing sectors is the manufacturing industry. Therefore, the manufacturing industry is required to minimize carbon emissions. One of the efforts to solve the emission problem is to minimize machine downtime throughout the production procedure, which stands for no-idle permutation flowshop scheduling (NIPFSP). This article uses two metaheuristic algorithms, giant trevally optimizer (GTO) and African vultures optimization algorithm (AVOA), to solve the carbon emission problem. Both algorithms are tested on 3 cases with 30 runs for every population and iteration. To compare the outcomes of each algorithm, an independent sample t-test was employed. The results show that the GTO algorithm has better results than the AVOA algorithm on small and large case data. The findings indicate that both the GTO and AVOA algorithms yield comparable results when applied to medium-sized research datasets, suggesting their effectiveness in such scenarios.

This is an open access article under the [CC BY-SA](#) license.



Corresponding Author:

Cantika Febrita

Department of Industrial Engineering, Faculty of Engineering, Muhammadiyah Malang University

Tlogomas highway no 246 Malang, East Java, Indonesia

Email: cantikafebritaa@gmail.com

1. INTRODUCTION

The high use of carbon emissions has become a concern in the world, and one of the sectors that contribute the highest carbon emissions is the manufacturing industry [1]. According to research [2], high carbon emissions can be caused by high energy consumption from machine operation [3], and the energy-efficiency problems in the manufacturing sector are closely related to scheduling [4]. There are two practical techniques to reduce carbon emissions in manufacturing: using energy-efficient equipment [5] and implementing an optimal low-carbon production strategy [6]. In the carbon-efficient scheduling model proposed by Ding *et al.* [7], carbon emissions are only related to energy consumption but ignore carbon emissions caused by preparation time and idle time on machines [8]. Furthermore, Asmatulu *et al.* [9] showed that when idle machines can consume a large quantity of energy, the manufacturing process can reduce carbon emissions by minimizing idle machines [10]. One exciting solution is to use no-idle machine scheduling to minimize carbon emissions. This approach is much more effective in reducing carbon emissions in the manufacturing process [7].

No-idle permutation flow shop problem (NIPFSP) is different from other permutation flow shop problems (PFSP) because, in NIPFSP, each machine must process jobs without idle time from the start of the job until the last job is completed [11]. NIPFSP is categorized as a constraint problem [12]. This is because the NIPFSP involves constraints that must be satisfied to find a feasible solution. For example, in the

NIPFSP, there is a constraint that no-idle time is allowed between the processing of jobs on different machines. This constraint must be satisfied in order to find a feasible solution to the problem. In real life, NIPFSP has been applied in the production process of glass fiber processing [13]. Then various methods were introduced by several researchers such as heuristic algorithms [14], to solve NIPFSP. There is also research on Flow Shop Dependent Sequence Setup scheduling using the ant-lion optimizer algorithm to minimize carbon emissions [15].

Some research on NIPFSP uses several algorithms. However, NIPFSP research aimed at minimizing carbon emissions has not been carried out, so NIPFSP research is needed to minimize carbon emissions. Thus, the metaheuristic algorithm can be applied in solving NIPFSP to minimize carbon emissions. Due to the absence of such research, this research is exciting to be researched further by presenting new algorithms, namely giant trevally optimizer (GTO) and African vultures optimization algorithm (AVOA), to solve the NIPFSP problem to minimize carbon emissions. GTO is a metaheuristic algorithm that mimics giant trevally fish's behavior when hunting seabirds [16]. Furthermore, AVOA mimics the behavior of African vultures when hunting prey [17]. GTO is proposed because it is proven to perform problem-solving for global optimization, capable of solving optimization problems and sustainable global engineering design [16]. However, GTO has never been used to solve the NIPFSP problem. Whereas AVOA has also solved various problems such as shell and tube heat exchanger optimization [18], membrane fuel cell optimization [19], and multi-objective flexible job shop scheduling problems [20]. However, AVOA has never been used to minimize carbon emissions.

From the aforementioned explanation, it can be inferred that the NIPFSP issue in minimizing carbon emissions needs additional research using the most recent algorithms. This research proposes the GTO and AVOA algorithms because they are novel and have never been applied to solve NIPFSP problems focusing on minimizing carbon emissions. Thus, this research aims to implement two of the newly developed algorithms, GTO and AVOA, to solve the NIPFSP problem by minimizing carbon emissions and comparing the results of the GTO and AVOA algorithms on the NIPFSP problem to minimize carbon emissions.

2. DESCRIPTION OF NIPFSP

2.1. Assumptions and mathematical formula

The terminology and mathematical model formulation utilized in the NIPFSP problem are described in this section. Machines can be idle after finishing work in a typical PFSP scenario [21]. However, in NIPFSP, it is presumed that the machine must remain active and not idle once the tasks are accomplished. Completion time on the machine: job identification: machine identification: speed rate identification: The job processing time on the machine completes one job [22]. Researchers use several assumptions that will be used in this study. The assumptions used for NIPFSP scheduling include: i) the entire set of n jobs must be processed on the set of m machines in the same process order, ii) all processes arrive and are ready for processing at time 0, iii) to fulfill the no-idle requirement, the processing start time for the first job on each machine must be delayed, iv) every machine can only process one job and can only be processed on one machine at a certain time, v) when the first job starts processing, it cannot be interrupted until the completion time of the last job, vi) job processing time includes setup time, and vii) no-idle machines are allowed in between job processing. The notations used in the NIPFSP problem are shown in Table 1.

Table 1. The notation used in the NIPFSP problem

Notation	NIPFSP problem
m	Number of machines
n	Number of jobs
j	Index of the machine
i	Index of the job
S_j	Start time on machine j
$C_{i,j}$	Time required for completing the job i on machine j
F_j	Completion time of the j machine
$P_{i,j}$	Processing time of job i
$C_{i,j}$	Completion time on machine j
ϕ_j	Consumption of energy during the machine is idle
τ_j	Energy consumption during machine i is operating
C_{max}	Total time or completion time
TEC	Cumulative energy consumption
EC	Total carbon emission
Ef	Factor emission

The following is a mathematical formulation to minimize carbon emission in NIPFSP problems presented in the mixed integer programming (MIP) model:

Decision Variable

$$X_{i,j,r} = \begin{cases} 1, & \text{if job } i \text{ is the predecessor of job } k \\ 0, & \text{otherwise} \end{cases}$$

$$Y_{i,j,r} = \begin{cases} 1, & \text{If job } i \text{ is completed at speed } r \text{ on machine } j \\ 0, & \text{otherwise} \end{cases}$$

Objective purpose

$$\text{Min EC} = \text{TEC} * \text{Ef} \quad (1)$$

Constraint:

$$C_{i,1} \geq \sum_{r=1}^l P_{i1*} Y_{ilr} \quad \forall i = \{1, \dots, n\} \quad (2)$$

$$C_{ij} - C_{i,j-1} \geq \sum_{r=1}^l P_{i1*} Y_{ilr} \quad \forall j = \{2, \dots, m\}, i = \{2, \dots, n\} \quad (3)$$

$$C_{ij} - C_{kj} + DX_{ik} \geq \sum_{r=1}^l P_{i1*} Y_{ilr} \quad \forall j = \{1, \dots, m\}, i = \{1, \dots, n\}, k = \{1, \dots, n\} \quad (4)$$

$$C_{ij} - C_{kj} + DX_{ik} \leq D - \sum_{r=1}^l P_{i1*} Y_{ilr} \quad \forall j = \{1, \dots, m\}, i = \{1, \dots, n\}, k = \{1, \dots, n\} \quad (5)$$

$$C_{max} \geq C_{im} \quad \forall i = \{1, \dots, n\} \quad (6)$$

$$\sum_{r=1}^l Y_{ijr} = 1 \quad \forall i = \{1, \dots, n\}, j = \{1, \dots, m\} \quad (7)$$

$$Y_{ijr} = Y_{i,j+1,r} \quad \forall i = \{1, \dots, n\}, j = \{1, \dots, m\}, r = \{1, \dots, l\} \quad (8)$$

$$\theta_j = C_{max} - \sum_{i=1}^n \sum_{r=1}^l \frac{P_{i1} Y_{ilr}}{\mu_r} \quad \forall j = \{1, \dots, m\} \quad (9)$$

$$\text{TEC} = \sum_{i=1}^n \sum_{j=1}^m \sum_{r=1}^l \frac{P_{ij} \tau_j \lambda_r}{60 \mu_r} Y_{ijr} + \sum_{j=1}^m \frac{\varphi_j \theta_j \tau_j}{60} \quad (10)$$

$$S_j \leq C_{ij} - \sum_{r=1}^l \frac{P_{i1} Y_{ilr}}{\mu_r} \quad \forall i = \{1, \dots, n\}, j = \{1, \dots, m\} \quad (11)$$

$$F_j \geq C_{ij} \quad \forall i = \{1, \dots, n\}, j = \{1, \dots, m\} \quad (12)$$

$$F_j \geq S_j + \sum_{i=1}^n \sum_{r=1}^l \frac{P_{i1} Y_{ilr}}{\mu_r} \quad \forall i = \{1, \dots, n\}, j = \{1, \dots, m\} \quad (13)$$

Equation (1) represents the objective function for minimizing carbon emissions. The amount of time needed to finish each task on the first machine is depicted in Constraint (2). Constraint (3) guarantees that the subsequent operation will commence only if the preceding one has been finalized. The sequencing of every job is outlined in Constraints (4), (5). The makespan, or finishing time, is calculated in Constraint (6). Constraints (7) and (8) ensure that all machines process every job at the same machining.

3. RESEARCH METHOD

3.1. Giant trevally optimizer (GTO)

The first used algorithm is giant trevally optimizer (GTO) [16]. The GTO algorithm's foundation is the behavior displayed by diving giant trevallies when hunting seabirds. As a result, the GTO algorithm's optimization process comprises three distinct steps: a comprehensive search employing Levy flight, the selection of an area for hunting, and the pursuit and attack of prey by leaping out of the water. The positional representation of an individual within the GTO population is defined by (14).

$$X = \begin{bmatrix} X_1 \\ \vdots \\ X_i \\ \vdots \\ X_N \end{bmatrix} = \begin{bmatrix} x_{1,1} & \cdots & x_{1,j} & \cdots & x_{1,Dim} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ x_{1,1} & \cdots & x_{1,j} & \cdots & x_{1,Dim} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ x_{N,1} & \cdots & x_{N,j} & \cdots & x_{N,Dim} \end{bmatrix} \quad (14)$$

Step 1: Extensive search

Giant trevally fish are capable of covering considerable distances. The foraging movement pattern of giant trevally fish is simulated with (15):

$$X(t+1) = Best_p \times R + ((maximum - minimum) \times R + minimum) \times levy(Dim) \quad (15)$$

$X(t+1)$ represents the vector of position for the subsequent iteration giant gannets, The selection of currently available search space by the giant gannets is denoted by $Best_p$, which is based on the best position discovered during their previous search. The variable R represents a random number ranging from 0 to 1. The $Levy(Dim)$ term corresponds to a Levy flight, which is a specific type of non-Gaussian stochastic process. The step size in the Levy flight is determined by the Levy distribution [23].

One of the primary benefits of employing Levy flights is the ability to circumvent local optima, leading to enhanced convergence speed [24]. The calculation of $Levy(Dim)$ is performed according to (16). A step size of 0.01, referred to as "step," is utilized. The Levy flight distribution function index, denoted as β , is set to 1.5 in this particular investigation, adhering to the range of 0 to 2. Random variables u and v follow a normal distribution within the interval of 0 to 1. The computation of σ is determined by applying in (17).

$$levy(Dim) = Step \times \frac{u - \sigma}{|v|^{\frac{1}{\beta}}} \quad (16)$$

$$\sigma = \left(\frac{\Gamma(1+\beta) \times \sin(\frac{\pi\beta}{2})}{\Gamma(\frac{1+\beta}{2}) \times \beta \times 2^{\frac{\beta-1}{2}}} \right) \quad (17)$$

Step 2: Selecting an area

During the area selection step, the giant trevally employs a strategy to identify and choose the optimal region within the selected search space, considering the abundance of food sources (seabirds) available for hunting prey. This behavior is mathematically simulated by (18).

$$X(t+1) = Best_p \times A \times R + Mean_Info - X_i(t) \times R \quad (18)$$

Equation (19) calculates the $Mean_Info$, which represents the mean value obtained from utilizing all accessible information from prior points. $X_i(t)$ denotes the i -th position of the giant trevally at the latest iteration, while $X(t+1)$ signifies the location vector of the giant trevally for the subsequent iteration. A , a parameter controlling the position change, ranges between 0.3 and 0.4.

$$Mean_info = \frac{1}{N} \sum_{i=1}^N X_i(t) \quad (19)$$

Step 3: Attack

During the GTO algorithm's exploitation (intensification) step, the goby engages in chasing and attacking its prey, the bird. This pursuit involves the goby getting close to the bird and executing an acrobatic leap out of the water to catch it. To replicate this behavior, the GTO algorithm incorporates the concept of visual distortion caused by light refraction, which occurs when light waves cross the boundary between different media such as glass, air, and water.

Based on [25], at the refractive point, both the incident and refracted rays must form an angle with the normal line to the surface. The medium through which light rays pass is equally essential. Snell's law clarifies this relationship using a refractive index, a fixed value for a given medium. Here, if we know the angle of incidence, we can predict the refractive angle. Likewise, if we are aware of the refractive angle, the incidence angle is predictable. Snell's law is shown in (20). Where $\eta_1=1.00029$ and $\eta_2=1.33$ are the precise refractive indices of water and air, respectively, and 1 and 2 are the incidence and refractive angles. 2 is the arbitrary range between (0 and 360). The (11) can be determined using (19). Following that, the visual distortion V can be calculated using (22).

$$\eta_1 \sin \theta_1 = \eta_2 \sin \theta_2 \quad (20)$$

$$\sin \theta_1 = \frac{\eta_1}{\eta_2} \sin \theta_2 \quad (21)$$

$$V = \sin(\theta_1^\circ) \times \mathcal{D} \quad (22)$$

The calculation of sin involves taking the sine of the variable in degrees. The distance between the prey and the attacker denoted as \mathcal{D} , is determined using (23). In this equation, $Best_p$ represents the best solution obtained, which denotes the quarry location. The action of the giant trevally during the pursuit and leaping out of the water is a mathematical simulation by (24). The solution for the next iteration, $X(t+1)$, is generated through the attack step. The launch velocity, \mathcal{L} , used to mimic the bird chase, can be calculated using (25). In this equation, $F_{obj}(X_i(t))$ denotes X 's fitness value as the latest iteration, (t) . The jump slope function, \mathcal{H} , in (23) facilitates the adaptive transition from the exploration phase to the exploitation phase, and its calculation is determined by (26).

$$\mathcal{D} = |(Best_p - X_i(t))| \quad (23)$$

$$X(t+1) = \mathcal{L} \times \mathcal{V} \times \mathcal{H} \quad (24)$$

$$\mathcal{L} = X_i(t) \times \sin(\theta_1^\circ) \times F_{obj}(X_i(t)) \quad (25)$$

$$\mathcal{H} = R \times (2 - t \times \frac{2}{T}) \quad (26)$$

In the aforementioned context, the variables t and T respectively denote the latest iteration and the highest value of iterations. R represents a random value that signifies the giant trevally fish's specific motion sensing during the exploitation process. It is important to note that the value of \mathcal{H} decreases gradually from 2 to 0 as the iterations progress. During the exploitation step, the algorithm endeavors to exploit the solution environment, utilizing this decreasing trend of \mathcal{H} . Algorithm 1 shows the developed pseudocode of the GTO algorithm.

Algorithm 1: Pseudocode GTO

1. Begin
2. Enter a number for \mathcal{A} parameter
3. Indicate the number of giant trevallies; N
4. Define the termination condition, maximum iterations (T)
5. Create a Giant Trevally population (X) at random using (21)
6. for $t = 1: T$
7. apply LRV to change the best position to job sequences and calculate objective function $f(X)$ for each search agent
8. Classify the populace
9. Identify the greatest global solution ($Best_G$)
10. Identify $Best_p$ as the prey's position (best location)
11. for $i = 1: N$
12. Comprehensive Investigation Step
13. Using (19) and (20), compute the levy flight distribution function levy.
14. Determine new
15. If $f(Best_{NP}) < f(X(I, :))$
16. $X(I, :) = Best_{NP}$
17. If $f(Best_{NP}) < f(Best_p)$
18. $Best_G = Best_{NP}$
19. End if
20. End if
21. Selecting area step
22. Using (19), compute the mean of X
23. Using (18), Calculate $Best_{NP}$
24. Steps 15 – 20 should be repeated
25. Attacking move
26. Using (22), Determine distortion of visual \mathcal{V}
27. Determine the launch speed \mathcal{L} using (25)
28. Using (24), Determine $Best_{NP}$
29. The transition from exploration to exploitation using (26)
30. Steps 15 – 20 should be repeated
31. End for t
32. Best solution and visualization after the process

3.2. African vultures optimization algorithm (AVOA)

AVOA is a metaheuristic method developed by modeling and simulating African vulture foraging behavior [17]. Based on the four requirements outlined above, the AVOA technique can be divided into five stages in the foraging stage to emulate the behavior of different vultures.

Phase 1: Clustering

After the formation of the original population, the appropriateness of all solutions is selected in this step. The best vulture is acknowledged as the finest option. Using (27), the first, and second solution is regarded as the second-best vulture, while the third group is given the other vultures.

$$R(i) = \begin{cases} \text{BestVulture}_1 & \text{if } P_i = L_1 \\ \text{BestVulture}_2 & \text{if } P_i = L_2 \end{cases} \quad (27)$$

BestVulture_1 represents the best vulture, BestVulture_2 represents the second-best, L_1 and L_2 are two random numbers between (0.1) and the sum is 1. P_i is used in (28) through the roulette-wheel approach. Here, F_i represents the fitness of the first and second groups of vultures, and n is the total number of the two vultures.

$$P_i = \frac{F_i}{\sum_{i=1}^n F_i} \quad (28)$$

Phase 2: Starvation rate of vultures

Assume a bunch of vultures is not hungry. In that situation, they have enough energy to travel further distances to find food. However, if they are hungry, they will not be able to preserve their long-distance travel. As a result, hungry vultures will exhibit hostile behavior. This behavior can be calculated using (29), which indicates the vultures' shift from exploration to exploitation. F indicates that the vultures are full, rand_i is a variable with a number that fluctuates between (0.1), t is computed by (30), and z is a random number in the range (1.1) that changes with every repetition.

$$F = (2 \times \text{rand}_i + 1) \times z \times \left(1 - \frac{\text{iteration}_i}{\text{maxiteration}} \right) + t \quad (29)$$

$$t = h \times \left(\sin^w \left(\frac{\pi}{2} \times \frac{\text{iteration}_i}{\text{maxiteration}} \right) + \left(\cos^w \left(\frac{\pi}{2} \times \frac{\text{iteration}_i}{\text{maxiteration}} \right) - 1 \right) \right) \quad (30)$$

The predefined parameter w determines the probability of a vulture performing the exploitation stage. In addition, the latest value iteration_i is the number of total iterations, h is a random value between denoted as iteration, and h is a random value between values (-2.2). Based on (29), F will steadily decrease as the number of iterations increases. When the number of $|F_i|$ exceeds one, vultures enter the stage of exploration and search for new food in diverse locales. Otherwise, vultures reach the exploitation stage, where they look for better food in the surrounding area.

Phase 3: Exploration phase

In AVOA, two alternative techniques are employed to inspect various random places, and a parameter designated $P1$ in the range of (0.1) is utilized to select one of the strategies. To determine one of the strategies during the phase, a random strategy $\text{rand } P_1$ between (0.1) is used. If the value is better or equal to the parameter $P1$ using in (31) in (32) is used if the value of rand_i is smaller than the parameter $P1$.

$$P(i+1) = R(i) - D(i) \times F_i \quad (31)$$

$$P(i+1) = R(i) - F_i + \text{rand}_2 \times ((ub - lb) \times \text{rand}_3 + lb) \quad (32)$$

$R(i)$ was selected as one of the finest vultures in the current iteration based on (27), F_i is the current iteration's vulture saturation rate determined in the most recent iteration based on (29), rand_2 is a random value between (0.1), and lb and ub are the variable's upper and lower bounds, respectively. rand_3 is accustomed to offering a high random coefficient on the scale of the search neighborhood to boost the variety and look for different sections of the search space. Equation (33) computes $D(i)$, which shows the distance between the current and optimal vultures. $P(i)$ denotes the i -th vulture's position, while X is a random number between (0.2).

$$D(i) = |X \times R(i) - P(i)| \quad (33)$$

Phase 4: Exploitation (first stage)

At the moment, the efficiency stage of AVOA is being explored. If $|Fi|$ is less than one, AVOA starts the first stage of exploitation. $P2$ is a parameter with a value between (0.1) that determines the chosen strategy. At the beginning of the first stage of exploitation, and $P2$ a random value between (0.1) is generated. In (34) illustrates this procedure.

$$P(i+1) = \begin{cases} D(i) \times (F_i + rand_4) - d(t) & \text{if } P_2 \geq rand_{p2} \\ R(i) - (S_1 + S_2) & \text{if } P_2 < rand_{p2} \end{cases} \quad (34)$$

$rand_4$ is a random number between (0.1), and $d(t)$ is the separation between the vultures and one of the two groups' best vultures, as calculated by (35). S_1 and S_2 are determined using (36) and (37), respectively. Where, $rand_5$ and $rand_6$ are random values between (0.1), respectively.

$$d(i) = R(i) - P(i) \quad (35)$$

$$S_1 = R(i) \times \left(\frac{rand_5 \times P(i)}{2\pi} \right) \times \cos(P(i)) \quad (36)$$

$$S_2 = R(i) \times \left(\frac{rand_6 \times P(i)}{2\pi} \right) \times \sin(P(i)) \quad (37)$$

Phase 5: Exploitation (second stage)

This phase of the algorithm is conducted if $|Fi|$ is smaller than 0.5. $rand_3$ is produced in the range of (0.1) at the start of this phase. Thus, if P_3 is higher than or equal to $rand_3$, the approach is to attract different vultures to the food source, resulting in competitive behavior. As a result, (38) can be used to update the vultures' position. Equations (39) and (40) are used to calculate A_1 and A_2 .

$$P(i+1) = \frac{A_1 + A_2}{2} \quad (38)$$

$$A_1 = BestVulture_1 - \frac{BestVulture_1(i) \times P(i)}{BestVulture_1(i) - (P(i))^2} \times F_i \quad (39)$$

$$A_2 = BestVulture_2 - \frac{BestVulture_2(i) \times P(i)}{BestVulture_2(i) - (P(i))^2} \times F_i \quad (40)$$

As AVOA progresses to the second stage, to scavenge for leftovers, vultures congregate around the best vulture. As a consequence, (41) can be used to update the vultures' position. In this case, D represents the problem dimension, and the AVOA efficacy is improved by employing the *Lévy flight (LF)* pattern, which is calculated using in (42). Where, v and u are each random values between (0.1), and β is a fixed number of 1.5. Algorithm 2 shows the development of the AVOA algorithm.

$$P(i+1) = R(i) - |d(t)| \times F_i \times \text{levy}(d) \quad (41)$$

$$LF(x) = 0,001 \times \frac{ux \sigma}{|v|^{\frac{1}{\beta}}} \quad (42)$$

$$\sigma = \left(\frac{\Gamma(1+\beta) \times \sin(\frac{\pi\beta}{2})}{\Gamma(1+\beta/2) \times \beta \times 2 \times (\frac{\beta-1}{2})} \right)^{\frac{1}{\beta}} \quad (43)$$

Algorithm 2: AVOA

1. The population size N and the maximum number of iterations T are inputs.
2. Outputs: The location of the vulture and its fitness value
3. Start the random population P_i ($i = 1, 2, \dots, N$)
4. do while (no halting condition satisfied)
5. Use LRV to alter the best position in job sequences.
6. Vulture carbon emission values
7. Make $P_{BestVulture1}$, the Vulture's location (Best Vulture Category 1 first best location).
8. Make $P_{BestVulture2}$, the Vulture's position (the second best location in Best Vulture Category 2).
9. do for (each vulture (P_i))

```

10. Using Equation (27) determine R(i)
11. Use Equation (34) to update the F
12. if ( $|F| \geq 1$ ) then
13.     if ( $P_1 \geq rand_{p1}$ ) then
14.         Update the location of the vulture using Equation (31)
15.     else
16.         Using Equation (32) update the location vulture
17. if ( $|F| < 1$ ) then
18.     if ( $|F| > 0.5$ ) then
19.         if ( $P_2 \geq rand_{p2}$ ) then
20.             Using Equation (34) Update the location vulture
21.         else
22.             Using Equation (35) Update the location vulture
23.     else
24.         if ( $P_3 \geq rand_{p3}$ ) then
25.             Using Equation (39) Update the location vulture
26.         else
27.             Using Equation (42) Update the location vulture
28. Return  $P_{BestVulture1}$ 

```

3.3. Experimental data and procedures

This research uses 3 different case studies and energy emissions per kWh of 0.998 kg in all 3 case studies. In the first case study [26] with a problem of 10 jobs 6 machines, the second case study [27] with a problem of 50 jobs 6 machines, and the third case study [28] with a problem of 100 jobs 10 machines. The experimental procedure in both algorithms is run 30 times to find the minimum energy consumption. Each case study consists of two variations of iterations (100 and 200) and population (100 and 200) in each algorithm. To determine which method is superior and more successful between AVOA and GTO, an independent test was used to examine the sig value (2-tailed). When the value is > 0.05 , then GTO and AVOA algorithms perform similarly. When the value is < 0.05 , the GTO and AVOA algorithms behave differently. The results of the experiment will also be presented in a boxplot diagram. This experiment was carried out using a Windows 10 device with an Intel Core-i5 processor.

4. RESULT AND DISCUSSION

This section describes and explains the study's results by considering the effect of different population numbers and the number of iterations on carbon emission consumption. In addition, this section will compare the effectiveness of the GTO and AVOA algorithms with the independent sample t-test. The following are the results and discussion of the AVOA and GTO algorithms.

Table 2 shows the comparison of populations and iteration experiment results using the GTO and AVOA algorithms. In Case 1, the best experimental result is at iteration 100, population 200. The most effective result in Case 2 is at iteration 200 population 200. The most effective result in Case 3 is at iteration 100 population 100. According to the experimental results, the larger the population and iteration, the lower the carbon emission consumption. And, conversely, as the population and iterations decrease, so do the carbon emissions produced. Table 3 displays the outcomes of running EC with the GTO and AVOA algorithms. The results reveal that the EC in Cases 1, 2, and 3 have different values. Case 1 demonstrates that the GTO and AVOA algorithms have the same variation in data distribution. Case 2 demonstrates that the GTO data distribution is more diversified than the AVOA data distribution. In example 3, the GTO data distribution is smaller than the AVOA data distribution.

Table 2. Results of iteration and population comparison experiments utilizing the GTO and AVOA algorithms

Research	Number of machines and jobs	Iteration	Population	GTO EC consumption	AVOA EC consumption
Case 1	10 jobs and 6 machines	100	100	16833.58	16868.96
			200	16833.58	16867.14
		200	100	16833.58	16867.14
			200	16833.58	16868.96
Case 2	50 jobs and 10 machines	100	100	11988.12	11996.24
			200	11995.99	11995.29
		200	100	11993.37	11995.34
			200	11990.34	11995.73
Case 3	100 jobs and 10 machines	100	100	2068.451	2226.769
			200	2069.28	2228.185
		200	100	2069.014	2235.167
			200	2069.521	2222.163

Table 3. Test result carbon emission consumption of GTO and AVOA algorithms

Cases	Result	GTO	AVOA
Case 1	Average	16833.585	16883.517
	Standard Deviation	0.000	19.543
	Minimum	16833.590	16867.136
	Maximum	16833.590	16930.437
Case 2	Average	11995.077	11995.533
	Standard Deviation	4.207	1.505
	Minimum	11990.245	11992.212
	Maximum	12002.951	11998.288
Case 3	Average	2069.239	2237.446
	Standard Deviation	0.245	21.524
	Minimum	2068.635	2198.696
	Maximum	2069.656	2278.357

Table 4. Results of the independent sample t-test

	Case 1		Case 2		Case 3	
	AVOA	GTO	AVOA	GTO	AVOA	GTO
Mean	16883.517	16833.59	11995.53	11995.08	2237.446	2069.239
Standard Deviasi	19.54305	0	1.505	4.207	21.524	0.245
t	-13.676		0.559		42.799	
Sig.(2-tailed)	0		0.578		0	

5. CONCLUSION

This study proposes GTO and AVOA to minimize carbon emissions. To answer the NIPFSP problem, these techniques are compared. The experimental results reveal that in case 1, with small data, the GTO method is more effective at minimizing carbon emissions. In Case 2 which is medium data, AVOA and GTO algorithms are effectively used to minimize carbon emissions, and In Case 3, the GTO method is utilized more efficiently to minimize carbon emissions, which is a large amount of data in the NIPFSP problem. The parameter testing of the GTO and AVOA algorithms reveals that the higher the iteration and population, the less carbon is consumed. Suggestions for future research are to compare with other algorithms with different objectives and approaches.

ACKNOWLEDGEMENTS

The authors would like to thank the Department of Industrial Engineering's Optimisation Laboratory for allowing us to use their facilities.




REFERENCES

- [1] D. M. Utama and M. D. Primayesti, "A novel hybrid Aquila optimizer for energy-efficient hybrid flow shop scheduling," *Results in Control and Optimization*, vol. 9, p. 100177, Dec. 2022, doi: 10.1016/j.rico.2022.100177.
- [2] C. Li, Y. Tang, L. Cui, and P. Li, "A quantitative approach to analyze carbon emissions of CNC-based machining systems," *Journal of Intelligent Manufacturing*, vol. 26, no. 5, pp. 911–922, Oct. 2015, doi: 10.1007/s10845-013-0812-4.
- [3] D. M. Utama, A. A. P. Salim, and D. S. Widodo, "A novel hybrid archimedes optimization algorithm for energy-efficient hybrid flow shop scheduling," *International Journal of Advances in Intelligent Informatics*, vol. 8, no. 2, pp. 237–250, Jul. 2022, doi: 10.26555/ijain.v8i2.724.
- [4] D. M. Utama, M. D. Primayesti, S. Z. Umamy, B. M. N. Kholifa, and A. D. Yasa, "A systematic literature review on energy-efficient hybrid flow shop scheduling," *Cogent Engineering*, vol. 10, no. 1, Dec. 2023, doi: 10.1080/23311916.2023.2206074.
- [5] S. Rahimifard, Y. Seow, and T. Childs, "Minimising Embodied Product Energy to support energy efficient manufacturing," *CIRP Annals*, vol. 59, no. 1, pp. 25–28, 2010, doi: 10.1016/j.cirp.2010.03.048.
- [6] C. Zhang, P. Gu, and P. Jiang, "Low-carbon scheduling and estimating for a flexible job shop based on carbon footprint and carbon efficiency of multi-job processing," *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture*, vol. 229, no. 2, pp. 328–342, Feb. 2015, doi: 10.1177/0954405414527959.
- [7] J. Y. Ding, S. Song, and C. Wu, "Carbon-efficient scheduling of flow shops by multi-objective optimization," *European Journal of Operational Research*, vol. 248, no. 3, pp. 758–771, Feb. 2016, doi: 10.1016/j.ejor.2015.05.019.
- [8] W. Lin, G. Tian, Z. Li, Y. Zhang, and C. Zhang, "Flow shop scheduling with low carbon emission and variable machining parameters," *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture*, vol. 233, no. 5, pp. 1561–1572, Apr. 2019, doi: 10.1177/0954405418782290.
- [9] R. Asmatulu, W. Khan, and M. B. Yildirim, "Acoustical properties of electrospun nanofibers for aircraft interior noise reduction," *ASME International Mechanical Engineering Congress and Exposition, Proceedings*, 2010.
- [10] G. Mouzon, M. B. Yildirim, and J. Twomey, "Operational methods for minimization of energy consumption of manufacturing equipment," *International Journal of Production Research*, vol. 45, no. 18–19, pp. 4247–4271, Sep. 2007, doi: 10.1080/00207540701450013.
- [11] F. Zhao, L. Zhang, Y. Zhang, W. Ma, C. Zhang, and H. Song, "A hybrid discrete water wave optimization algorithm for the no-idle flowshop scheduling problem with total tardiness criterion," *Expert Systems with Applications*, vol. 146, p. 113166, May




- 2020, doi: 10.1016/j.eswa.2019.113166.
- [12] M. Fatih Tasgetiren, H. Öztol, L. Gao, Q. K. Pan, and X. Li, "A variable iterated local search algorithm for energy-efficient no-idle flowshop scheduling problem," *Procedia Manufacturing*, vol. 39, pp. 1185–1193, 2019, doi: 10.1016/j.promfg.2020.01.351.
 - [13] P. J. Kalczynski and J. Kamburowski, "A heuristic for minimizing the makespan in no-idle permutation flow shops," *Computers and Industrial Engineering*, vol. 49, no. 1, pp. 146–154, Aug. 2005, doi: 10.1016/j.cie.2005.05.002.
 - [14] R. K. Chakraborty, A. Abbasi, and M. J. Ryan, "Multi-mode resource-constrained project scheduling using modified variable neighborhood search heuristic," *International Transactions in Operational Research*, vol. 27, no. 1, pp. 138–167, Jan. 2020, doi: 10.1111/itor.12644.
 - [15] D. S. Widodo and D. M. Utama, "The hybrid ant lion optimization flow shop scheduling problem for minimizing completion time," *Journal of Physics: Conference Series*, vol. 1569, no. 2, p. 022097, Jul. 2020, doi: 10.1088/1742-6596/1569/2/022097.
 - [16] H. T. Sadeeq and A. M. Abdulazeez, "Giant trevally optimizer (GTO): A novel metaheuristic algorithm for global optimization and challenging engineering problems," *IEEE Access*, vol. 10, pp. 121615–121640, 2022, doi: 10.1109/ACCESS.2022.3223388.
 - [17] B. Abdollahzadeh, F. S. Gharehchopogh, and S. Mirjalili, "African vultures optimization algorithm: A new nature-inspired metaheuristic algorithm for global optimization problems," *Computers and Industrial Engineering*, vol. 158, p. 107408, Aug. 2021, doi: 10.1016/j.cie.2021.107408.
 - [18] D. Gürses, P. Mehta, S. M. Sait, and A. R. Yildiz, "African vultures optimization algorithm for optimization of shell and tube heat exchangers," *Materialprüfung/Materials Testing*, vol. 64, no. 8, pp. 1234–1241, Aug. 2022, doi: 10.1515/mt-2022-0050.
 - [19] Y. Chen and G. Zhang, "New parameters identification of Proton exchange membrane fuel cell stacks based on an improved version of African vulture optimization algorithm," *Energy Reports*, vol. 8, pp. 3030–3040, Nov. 2022, doi: 10.1016/j.egyr.2022.02.066.
 - [20] Z. He, B. Tang, and F. Luan, "An improved African vulture optimization algorithm for dual-resource constrained multi-objective flexible job shop scheduling problems," *Sensors*, vol. 23, no. 1, p. 90, Dec. 2023, doi: 10.3390/s23010090.
 - [21] C. Y. Cheng, S. W. Lin, P. Pourhejazy, K. C. Ying, and Y. Z. Lin, "No-idle flowshop scheduling for energy-efficient production: An improved optimization framework," *Mathematics*, vol. 9, no. 12, p. 1335, Jun. 2021, doi: 10.3390/math9121335.
 - [22] X. S. Yang, T. O. Ting, and M. Karamanoglu, "Random walks, Levy flights, Markov chains and metaheuristic optimization," in *Lecture Notes in Electrical Engineering*, vol. 235 LNEE, 2013, pp. 1055–1064. doi: 10.1007/978-94-007-6516-0_116.
 - [23] D. M. Utama, A. K. Garside, and W. Wicaksono, "Development of a Three-Stage Hybrid Flowshop Algorithm Considering Setup Time in Indonesian," *Jurnal Ilmiah Teknik Industri*, vol. 18, no. 1, pp. 72–78, Jul. 2019, doi: 10.23917/jiti.v18i1.7683.
 - [24] M. Chawla and M. Duhan, "Levy flights in metaheuristics optimization algorithms—a review," *Applied Artificial Intelligence*, vol. 32, no. 9–10, pp. 802–821, Nov. 2018, doi: 10.1080/08839514.2018.1508807.
 - [25] C. A. Bennett, *Principles of Physical Optics*. usa: Hoboken, NJ, 2022.
 - [26] J. Carlier, "Scheduling with disjunctive constraints," *RAIRO Recherche Operationnelle*, vol. 12, no. 4, pp. 333–350, Mar. 1978, doi: 10.1051/ro/1978120403331.
 - [27] C. R. Reeves, "A genetic algorithm for flowshop sequencing," *Computers and Operations Research*, vol. 22, no. 1, pp. 5–13, Jan. 1995, doi: 10.1016/0305-0548(93)E0014-K.
 - [28] J. Heller, "Some numerical experiments for an $M \times J$ flow shop and its decision-theoretical aspects," *Operations Research*, vol. 8, no. 2, pp. 178–184, Apr. 1960, doi: 10.1287/opre.8.2.178.

BIOGRAPHIES OF AUTHORS



Dana Marsetiya Utama    is currently a Lecturer and a Researcher in the Industrial Engineering Department, at the University of Muhammadiyah Malang Indonesia. His research interests include optimization engineering, scheduling, and modeling. He received a bachelor's degree in industrial engineering from Trunojoyo University, in 2008, and a master's degree in industrial engineering from the University of Brawijaya, in 2011. He can be contacted at email: dana@umm.ac.id



Cantika Febrita    is currently a Bachelor's Student in the Industrial Engineering Department, at the University of Muhammadiyah Malang Indonesia. Her research interests include optimization engineering, scheduling, and modeling. She can be contacted at email: cantikafebrita@gmail.com